AN EMPIRICAL EVALUATION OF CONVERGENCE TO CORRELATED EQUILIBRIA: INTRODUCING MULTI-STAGE MULTIPLICATIVE-WEIGHTS UPDATE

MICHAEL HAN AND ASHLEY YU

Abstract. No-regret learning algorithms are an important component of advances in solving large-scale games. These algorithms are commonly used to solve games such as Diplomacy, an AI benchmark with a large action space where agents compete to dominate a map of Europe. We introduce Multi-Stage Multiplicative-Weights Update (MS-MWU), which shows an improvement upon existing external-regret minimizing algorithms such as MWU across all our experiments. We also perform an empirical evaluation of classic no-regret algorithms such as Multiplicative-Weights Update (MWU) and Optimistic Multiplicative-Weights Update (OMWU). Furthermore, we test swap regret minimization algorithms such as the no swap-regret algorithm of Blum & Mansour (2007) and the TreeSwap algorithm of Dagan et al (2024). We play these algorithms against each other and randomized adversaries on hundreds of subgames of Diplomacy along with Kuhn Poker and random games. Across all these games, our experiments show that MS-MWU converges significantly faster than MWU/OMWU. We experimentally show that swap regret and external regret remain very similar at all iterations. In other words, external regret minimization algorithms such as MWU outperform swap regret minimization algorithms such as BM in terms of rate of convergence and time complexity, even for very large time horizons.

1. Introduction

No-regret learning provides a principled framework for repeated decision-making and multi-agent interactions. In this setting, agents repeatedly choose actions and receive payoffs or losses, adjusting their strategies over time based on observed feedback. The fundamental objective is to ensure that an agent's performance, as measured by its cumulative payoff, is competitive with the best single action in hindsight. When the agent's *regret* (the difference between these two quantities) grows sublinearly in the total number of rounds, the algorithm is said to be no-regret.

Within no-regret learning, external regret is one of the most commonly studied variants. By ensuring that the agent does nearly as well as any single fixed action, external regret minimization drives repeated games toward a coarse correlated equilibrium (CCE). CCE can be viewed as a natural generalization of the classical Nash equilibrium, allowing for correlated strategies recommended by an external mediator. Nonetheless, while coarse correlated equilibria are appealing for their generality and relative tractability, some game-theoretic analyses focus on *swap regret*, which requires the learner to compete against the best function mapping from its actual played actions to alternative actions. Minimizing swap regret guarantees convergence to a correlated equilibrium (CE), offering a tighter solution concept than a CCE but often demanding more involved algorithms.

1.1. Challenges with complex action spaces.

A variety of regret-minimizing methods for external and swap regret have been proposed in the literature. Among the most notable

are the Multiplicative-Weights Update (MWU) (Nisan et al. (2007)) algorithm and its optimistic variant, Optimistic Multiplicative-Weights Update (OMWU) (Syrgkanis et al. (2015)), and the improved algorithm, Kernelized OMWU (KOMWU)(Farina et al. (2022)), which are widely used for external regret. For swap regret, classic approaches include the algorithm of Blum and Mansour (2007) (Blum and Mansour (2007)) and the more recent TreeSwap (Dagan et al. (2024)) algorithm by Dagan et al. (2024) and Multi-Scale MWU algorithm by Peng et al. (Peng and Rubinstein (2023)). While these methods provide important theoretical guarantees, practitioners often encounter slow convergence or high computational costs when the action space is very large or the strategic structure is complex (Daskalakis et al. (2023)). This limitation is particularly evident in challenging environments such as Diplomacy, and even in simpler settings like Kuhn Poker or randomly generated games.

1.2. **Main Result: Multi-Stage Multiplicative-Weights Update.** In response to the challenges regarding large action spaces, we propose a new external-regret minimizing algorithm called Multi-Stage Multiplicative-Weights Update (MS-MWU). Our approach partitions the time horizon into multiple stages, where each stage uses a carefully tuned learning rate and transition criterion. This staged framework mitigates the drawbacks of a single fixed-rate update, allowing the agent to adapt more effectively to changing payoff structures or adversarial conditions. In our experiments, MS-MWU consistently converges faster than MWU and OMWU across Diplomacy subgames, Kuhn Poker, and random games, even for very large time horizons. Moreover, we observe that while swap regret guarantees a stronger equilibrium concept in theory, external-regret algorithms such as MWU often converge at a similar or faster rate in practice, suggesting that stronger theoretical guarantees do not always translate into better empirical performance.

To provide intuition for why MS-MWU achieves sublinear regret with improved convergence, we offer a sketch of its proof. We begin by dividing the total number of rounds, T, into S stages, each stage operating for T_s rounds with a fixed learning rate η_s. Within each stage, we apply a multiplicative-weights update rule that guarantees a regret bound proportional to $\sqrt{T_s \log N}$, where N is the number of available actions. Between stages, we adjust η_s based on observed performance, effectively allowing the algorithm to "reset" or recalibrate. Summing over all stages yields a total regret of $O(\sqrt{T \log N})$, matching standard MWU theoretical guarantees, but our staged approach tends to yield faster empirical convergence because it dynamically adapts to the problem's feedback structure and avoids overly aggressive or overly conservative updates.

In this paper, we introduce our new algorithm Multi-Stage Multiplicative-Weights Update. We begin by establishing the preliminaries in section 2. We then provide a detailed explanation of our algorithm in section 3. In section 4, we provide our experimental results. Finally in section 5, we provide our conclusions and suggest future directions.

2. Preliminaries

2.1. **Game-Theoretic Setup.** A *normal-form* (or *strategic-form*) game represents a simultaneous-play setting in which each player chooses an action from a finite set without knowing the other players' choices. Formally, let $\mathcal{P} = \{1, 2, \ldots, n\}$ be the set of players. Each player $i \in \mathcal{P}$ has a finite action set A_i. A *joint action* is a tuple $a = (a_1, a_2, \ldots, a_n)$ with $a_i \in A_i$ for each i. Given a joint action a, each player i receives a *payoff* $u_i(a)$. The *normal-form game* is

then specified by

$$G \;=\; \Big(\mathcal{P},\, \{A_i\}_{i\in\mathcal{P}},\, \{u_i\}_{i\in\mathcal{P}}\Big).$$

Players may also randomize by selecting *mixed strategies*: a mixed strategy π_i for player i is a probability distribution over A_i. Let Σ_i denote the set of all such distributions, and $\Sigma = \prod_{i\in\mathcal{P}} \Sigma_i$. When a profile of mixed strategies $\pi = (\pi_1, \ldots, \pi_n)$ is played, the expected payoff to player i is $u_i(\pi) \;=\; \sum_{a\in A}\big(\prod_j \pi_j(a_j)\big)\, u_i(a)$.

Normal-form games can represent an enormous variety of strategic interactions, from classic two-player matrix games (like Rock–Paper–Scissors) to larger multi-player settings in domains such as online advertising, cybersecurity, or board/card games. Of particular interest are large-scale multi-agent problems such as *Diplomacy*, *Poker*, or *Avalon*, in which modern AI breakthroughs use computationally efficient algorithms from the field of *no-regret learning* to approximate equilibria.

2.2. **Equilibrium Concepts.** A *Nash equilibrium (NE)* in a normal-form game is a mixed strategy profile $\pi^* = (\pi_1^*, \ldots, \pi_n^*)$ such that no player can improve their expected payoff by unilaterally deviating to another strategy. Formally,

$$\max_{i\in\mathcal{P}}\Big(\max_{\hat{\pi}_i\in\Sigma_i} u_i\big(\hat{\pi}_i,\, \pi^*_{-i}\big) \;-\; u_i\big(\pi^*_i,\, \pi^*_{-i}\big)\Big) \;\le\; 0.$$

Finding a Nash equilibrium can be computationally hard, motivating the search for other, more tractable solution concepts.

A *correlated equilibrium* is a distribution p over joint actions $A = A_1 \times \cdots \times A_n$. A mediator draws an action profile $a = (a_1, \ldots, a_n)$ from p, then privately recommends action a_i to each player i. The distribution p constitutes a CE if no player can profitably deviate from the recommended action in expectation. Equivalently, for each i and any mapping $\phi : A_i \to A_i$,

$$\sum_{a\in A} p(a)\,\Big(u_i\big(\phi(a_i),\, a_{-i}\big) \;-\; u_i\big(a_i,\, a_{-i}\big)\Big) \;\le\; 0.$$

Because CE can be computed via linear programming, it is often easier to find than NE, especially in large games.

A *coarse correlated equilibrium* assigns a probability distribution p over joint actions but employs a weaker incentive constraint: each player must decide on a single alternative action before seeing their recommended action. Formally, for each i and each potential action $\hat{a}_i \in A_i$,

$$\sum_{a\in A} p(a)\,\Big(u_i\big(\hat{a}_i,\, a_{-i}\big) \;-\; u_i\big(a_i,\, a_{-i}\big)\Big) \;\le\; 0.$$

In summary, we have the inclusion

$$\text{Nash Equilibrium} \;\subseteq\; \text{Correlated Equilibrium} \;\subseteq\; \text{Coarse Correlated Equilibrium},$$

where each successive concept enlarges the set of possible equilibria. For two-player zero-sum games, these sets coincide in terms of payoffs due to the minimax theorem.

2.3. **Regret Minimization and Equilibrium Computation.** A primary technique for approximating equilibria in large or complex games is *no-regret learning*. Suppose that over T rounds, player i repeatedly chooses a mixed strategy π_i^t. An adversary (or environment) then assigns payoffs (or losses) for each action. The *external regret* of player i measures how much better they could have done by committing to a single best action in hindsight.

Formally, if

$$L_i^T \;=\; \sum_{t=1}^{T} u_i(\pi_i^t, \pi_{-i}^t) \quad \text{and} \quad L_i^T(a_i) \;=\; \sum_{t=1}^{T} u_i(a_i, \pi_{-i}^t),$$

then the external regret is

$$R_i^{E,T} \;=\; \max_{a_i \in A_i} L_i^T(a_i) \;-\; L_i^T.$$

An algorithm is *no-regret* if $R_i^{E,T} = o(T)$, ensuring that its average regret $\frac{1}{T} R_i^{E,T}$ goes to zero as $T \to \infty$. Crucially, when all players use no-external-regret strategies, their joint play converges to a coarse correlated equilibrium.

Stronger notions, such as *swap regret*, permit more flexible comparisons and yield convergence to a full correlated equilibrium if all players minimize swap regret. Although this notion is more powerful, algorithms that control swap regret can be more complex or slower in practice.

Algorithms like Multiplicative-Weights Update (MWU) and regret matching provide standard ways to achieve sublinear external (or internal) regret. In each round, MWU updates a "weight" for each action, reducing the weight of high-loss actions and increasing the probability of picking actions that have performed well historically. These methods have powered significant breakthroughs in large-scale game-solving, including poker variants, security settings, and board games like Diplomacy.

Despite the fact that MWU and other no-regret methods theoretically guarantee convergence to equilibrium, they can converge slowly in practice, especially in high-dimensional games with a large number of actions. This has sparked interest in refinements—such as the Multi-Stage Multiplicative-Weights Update (MS-MWU) algorithm we introduce—aimed at accelerating convergence while maintaining no-regret performance guarantees.

3. Multi-Stage Multiplicative-Weights Update

While classical Multiplicative-Weights Update (MWU) algorithms guarantee sublinear regret, they can exhibit slow convergence in practice, particularly for large or complex action spaces. In this section, we introduce *Multi-Stage Multiplicative-Weights Update* (MS-MWU), a variant of MWU that uses staged re-initialization to speed up convergence. The high-level idea is to divide the total time horizon into $\approx \sqrt{T}$ stages, update strategies within each stage using MWU, and then *reset* or *re-initialize* the weights based on cumulative performance at the end of each stage. By periodically re-initializing the algorithm's internal state, MS-MWU avoids becoming "stuck" with an outdated learning rate or misguided weight distribution when the environment changes or when certain actions prove much more profitable than others.

Algorithm 1 MS-MWU

Input: Number of actions N, time horizon T, decay rate r
Initialize: block size $M \approx \sqrt{T}$, $\eta = \sqrt{\frac{\log N}{M}}$, $P_{cum} = (0, \ldots, 0)$
for $t = 1$ to T **do**
  Normalize weights: $p_i^t = \frac{w_i^t}{\sum_{j=1}^{N} w_j^t}$ for all $i \in \{1, \ldots, N\}$
  Choose action: Randomly select action i with probability p_i^t
  Receive loss: ℓ_i^t for each action i
  Update weights: $w_i^{t+1} = w_i^t \cdot (1 - \eta)^{\ell_i^t}$ for each action i
  Accumulate strategies: $P_{cum} = P_{cum} + p^t$
  if $t \pmod M = 0$ **then**
    $w^t = \frac{P_{cum}}{M}$, $\eta = \frac{\eta}{r}$, $P_{cum} = 0$
  end if
end for

3.1. **Algorithm Description.**
Learning Rate and Block Updates. In each block of length M, we run a standard MWU procedure with a fixed learning rate η. At the end of the block, we re-initialize the weight vector w^t to be the average distribution encountered during that block. This ensures the algorithm "resets" to a representative strategy rather than fully preserving historical bias. Simultaneously, we reduce the learning rate η by a decay factor $r > 1$ to mitigate overshooting in later stages.

3.2. **Rationale and Advantages.** By partitioning the horizon into $\sqrt{T}$ stages, MS-MWU proactively combats the potential stagnation that arises from using a single global learning rate over the entire time horizon. Early stages, when losses and payoff structures may not yet be well understood, permit more exploratory behavior. As stages progress, the refined η and more informed "average distribution" help the algorithm quickly hone in on high-payoff actions.

Compared to vanilla MWU, which updates weights continuously without reset, MS-MWU stabilizes faster in many empirical settings (see Section 4). Even against optimistic variants (OMWU), staging often yields improved performance, especially in environments where loss vectors shift or where multiple actions become advantageous at different times. The key insight is that *accumulating strategies* and *averaging* across a block can jump-start the algorithm's distribution at each stage, preventing it from lingering on suboptimal actions or inefficient updates.

3.3. **Computational Complexity and Extensions.** The computational overhead of MS-MWU is comparable to MWU: each round requires $O(N)$ work to update weights and choose an action. The additional resetting steps at each stage involve re-initializing with a simple average, incurring negligible extra cost. Overall, MS-MWU retains the $O(T \cdot N)$ time complexity of MWU.

One can generalize MS-MWU by adjusting:

- **Block scheduling:** Instead of fixed-length blocks $M \approx \sqrt{T}$, employ adaptive stage lengths (e.g., halving M upon detecting slow convergence).
- **Adaptive decay:** Tune η non-uniformly depending on realized losses or regret levels within each block.

- **Hybrid approaches:** Combine MS-MWU with other advanced regret minimizers, such as OMWU or Regret Matching, for potentially tighter theoretical guarantees.

3.4. **Proof of Convergence Bound.** [MS-MWU Convergence] Let T be the total number of iterations and $M = \sqrt{T}$ for the MS-MWU algorithm. Then the MS-MWU updates achieve last-iterate convergence as $T \to \infty$. Furthermore, for finite but large T, choosing $M = AT^B$ with $B \in (0, 1)$ (close to 1) and a sufficiently large constant A ensures that the algorithm's error bound converges to 0 as $T \to \infty$.

Proof. First, recall that the standard Multiplicative Weights Update (MWU) algorithm has the property that its *average* probability distribution converges to an equilibrium as $T \to \infty$.

In our MS-MWU algorithm, we set $M = \sqrt{T}$. Over the *first* block of M iterations, as T grows, M also grows, which drives the regret in that block close to 0. For the subsequent blocks (each also of length M), the regret remains 0, so the probability distribution effectively stops changing, hence yielding *last-iterate convergence*. This behavior is observed empirically in small games such as Kuhn Poker.

Next, consider a large but finite T. We generalize the choice of M to

$$M \;=\; AT^B,$$

where A is a constant, and $B \in (0, 1)$ is close to 1. Choose A and B such that $T \leq C\,M$ for some sufficiently large constant C. In the *first* block of M iterations, we do not obtain the usual error bound $\epsilon = \sqrt{\frac{\ln N}{T}}$, but instead a somewhat larger error bound $\epsilon_2 = \sqrt{\frac{\ln N}{M}}$. In the next (at most) C blocks of length M, the standard MWU analysis ensures that the regret does not increase (further aided by the decay rate r). Hence the total error after these blocks is bounded by

$$\sqrt{\frac{\ln N}{M}} \times r^C.$$

Since M increases with T, this error bound decreases as T increases. Therefore, MS-MWU enjoys a theoretical error bound that converges to 0 as $T \to \infty$. □

3.5. **Summary.** MS-MWU modifies MWU to incorporate multi-stage resets, improving empirical convergence in diverse settings. The staged updates allow it to exploit patterns in each block of rounds while avoiding the risk of running with a single suboptimal rate across the entire time horizon. As Section 4 demonstrates, these design choices lead to faster regret reduction in practice, paving the way for more efficient approximate equilibria in large-scale or adversarial multi-agent scenarios.

4. Experimental Results

In this section, we present our empirical findings comparing classic no-regret algorithms (MWU, OMWU, Blum–Mansour, TreeSwap) to our proposed Multi-Stage Multiplicative-Weights Update (MS-MWU). Experiments were performed on both Kuhn Poker and on subgames of the Diplomacy environment.

Diplomacy is an extensive form game with normal-form subgames where there are 7 players and each chooses their action simultaneously. We used Gray et al. (2020)'s neural network, which scans and searches the 11-14 best actions for each power, and made modifications to obtain utility matrices. In total, we obtained over 500 2-player subgames for 6 pairs of

powers and the remaining 4 powers' actions randomly selected from the filtered actions. We also obtained several 7-player subgames.

Unless otherwise specified, each trial was repeated multiple times with different random seeds, and we plot the average external regret (or another convergence measure) over the number of iterations T.

4.1. **Kuhn Poker.** In figure 1, we measure external regret over T iterations. Notably, MWU consistently converges more quickly than BM. Although BM provides swap-regret guarantees theoretically, it incurs higher computational overhead in practice. Consequently, the simpler external-regret minimization approach (MWU) displays lower regret after relatively few rounds.

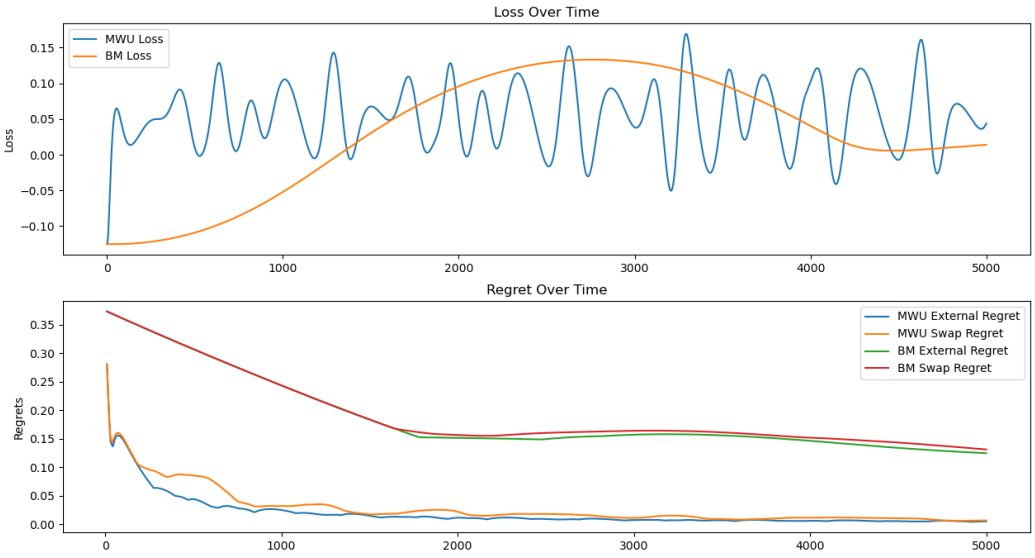

FIGURE 1. Kuhn Poker: MWU vs. Blum–Mansour. MWU converges faster, suggesting that external-regret minimization can be more efficient in practice than swap-regret algorithms like BM.

Figure 2 compares MWU and MS-MWU in Kuhn Poker. MS-MWU exhibits faster convergence throughout the entire learning horizon. This improvement is due to the staged re-initialization of strategy weights, which allows the algorithm to respond adaptively to changing payoffs and avoid getting stuck with suboptimal learning rates.

In Figure 3, we contrast MS-MWU with Optimistic MWU (OMWU). Although OMWU often outperforms standard MWU in adversarial settings, MS-MWU consistently achieves lower regret and converges more quickly. Intuitively, OMWU attempts to predict the next loss vector, whereas MS-MWU accumulates block-level statistics and re-initializes aggressively, resulting in more rapid adaptation.

4.2. **Diplomacy Subgames.** We next evaluate performance on selected two-player subgames extracted from the complex, multi-player game *Diplomacy*. These subgames have larger action spaces than Kuhn Poker, and they frequently involve more intricate payoff structures.

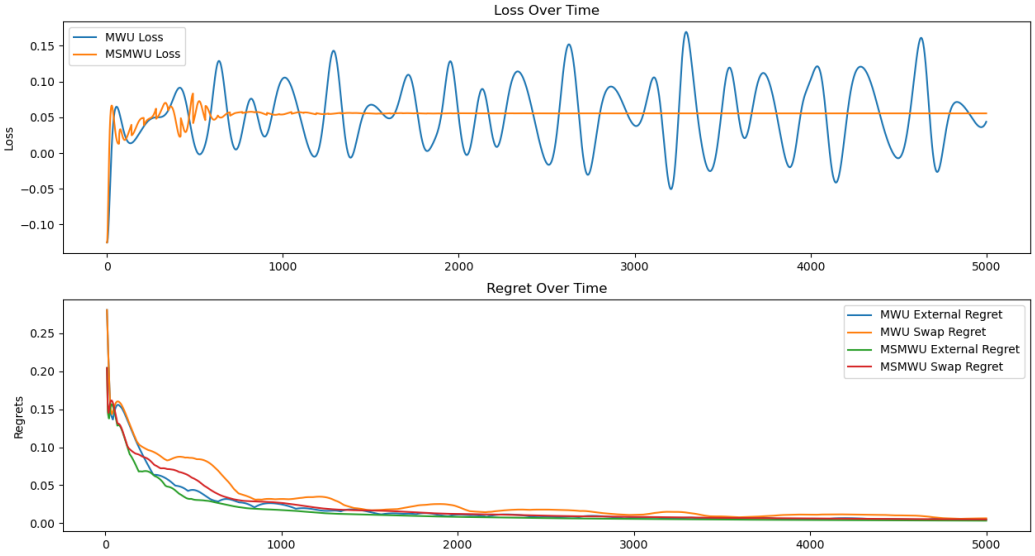

FIGURE 2. Kuhn Poker: MWU vs. MS-MWU. The proposed MS-MWU algorithm accelerates convergence by periodically re-initializing weights, outperforming vanilla MWU.

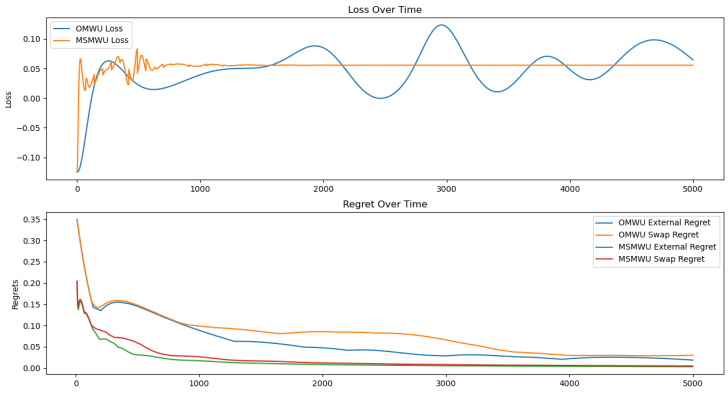

FIGURE 3. Kuhn Poker: OMWU vs. MS-MWU. Even compared to OMWU, MS-MWU shows a faster rate of convergence, highlighting the benefits of multi-stage updates.

4.2.1. *MWU vs. Blum–Mansour (BM).* Figure 4 shows a typical result comparing MWU and BM in a Diplomacy subgame. As with Kuhn Poker, MWU converges more quickly, suggesting that for practical computation of approximate equilibria in normal-form subgames, external-regret minimization is more efficient empirically than swap-regret minimization—even though swap-regret is theoretically stronger.

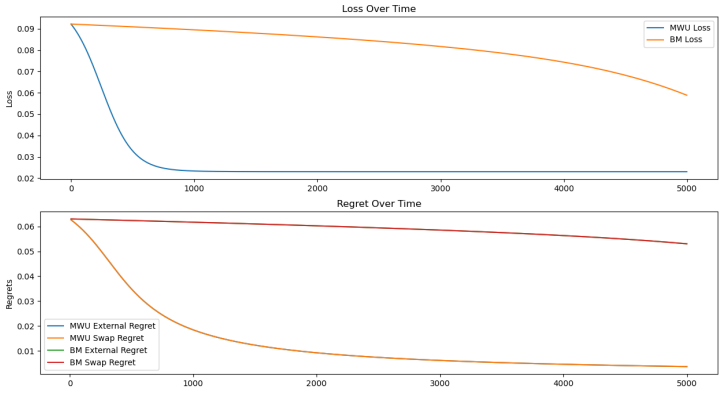

FIGURE 4. Diplomacy subgame: MWU vs. BM. MWU again demonstrates a faster decrease in regret, reinforcing the idea that simpler external-regret methods can be advantageous in practice.

4.2.2. *MWU vs. MS-MWU.* Finally, in Figure 5, we compare MWU to our MS-MWU algorithm on a Diplomacy subgame. The gap is even more pronounced here than in Kuhn Poker, indicating that MS-MWU's staged updates are particularly beneficial in environments with large, highly variable action spaces. Across a wide range of settings (different subgames, various payoff structures), MS-MWU consistently exhibited lower regret and faster convergence.

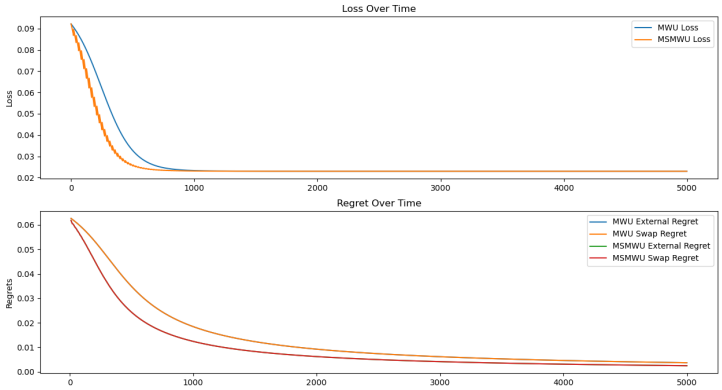

FIGURE 5. Diplomacy subgame: MWU vs. MS-MWU. Staging and adaptive re-initialization in MS-MWU lead to faster convergence compared to standard MWU in these larger, more complex subgames.

4.3. **Summary of Experimental Observations.** Overall, our experiments indicate three main takeaways:

- **Faster Convergence for MS-MWU.** The multi-stage strategy update consistently outperforms classical MWU/OMWU across both Kuhn Poker and Diplomacy subgames.

- **External vs. Swap Regret.** While swap-regret algorithms (e.g., BM) hold stronger theoretical guarantees, they often converge more slowly in practice than simpler external-regret algorithms like MWU and MS-MWU.
- **Scalability.** MS-MWU maintains strong performance even in large, complex payoff matrices, underscoring its potential for broader multi-agent AI applications where the action space and strategic complexity can be significant.

5. Conclusions and Further Research

We have presented a new no-regret learning algorithm, Multi-Stage Multiplicative-Weights Update (MS-MWU), designed to accelerate convergence in large-scale game-theoretic settings. By splitting the time horizon into stages and adaptively re-initializing strategy weights, MS-MWU enjoys empirical speedups over standard Multiplicative-Weights Update (MWU) and its optimistic variant (OMWU). Moreover, our results indicate that, although swap-regret algorithms such as Blum–Mansour and TreeSwap offer stronger theoretical guarantees for correlated equilibrium, they often converge more slowly in practice than simpler external-regret minimization methods.

Extensive experiments on Kuhn Poker and Diplomacy subgames confirm that MS-MWU reduces regret faster than baseline methods, thereby facilitating quicker approximate equilibrium computation. In both domains, the improvement is particularly noteworthy for large or complex action spaces, highlighting the practical advantages of staged updates. These findings underscore the potential for MS-MWU to be applied in broader multi-agent AI settings, where scalable equilibrium computation is vital.

Looking ahead, it remains an open theoretical question to establish tighter regret bounds for MS-MWU and to explore the algorithm's performance under more adversarial conditions or in multiplayer extensive-form games. Future work may also involve integrating MS-MWU with additional refinements—such as dynamic learning rates or advanced gradient-based optimizations—to further enhance its adaptability and robustness. We believe that continued research along these lines will help bridge the gap between strong theoretical guarantees and efficient performance in real-world multi-agent systems.

Acknowledgements

The authors would like to thank their excellent mentor, Noah Golowich, for his continued guidance and advice. The authors also would like to thank MIT PRIMES and its organizers for providing them with this incredible opportunity.

References

Blum, A. and Mansour, Y. (2007). From external to internal regret. *Journal of Machine Learning Research*, 8(6):1307–1324.

Dagan, Y., Daskalakis, C., Fishelson, M., and Golowich, N. (2024). From external to swap regret 2.0: An efficient reduction for large action spaces. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 1216–1222, New York, NY, USA. Association for Computing Machinery.

Daskalakis, C., Fishelson, M., and Golowich, N. (2023). Near-optimal no-regret learning in general games.

Farina, G., Lee, C.-W., Luo, H., and Kroer, C. (2022). Kernelized multiplicative weights for 0/1-polyhedral games: Bridging the gap between learning in extensive-form and normal-form games.

Gray, J., Lerer, A., Bakhtin, A., and Brown, N. (2020). Human-level performance in no-press diplomacy via equilibrium search. *arXiv preprint arXiv:2010.02923*.

Nisan, N., Roughgarden, T., Tardos, , and Vazirani, V. V., editors (2007). *Algorithmic Game Theory*. Cambridge University Press.

Peng, B. and Rubinstein, A. (2023). Fast swap regret minimization and applications to approximate correlated equilibria.

Syrgkanis, V., Agarwal, A., Luo, H., and Schapire, R. E. (2015). Fast convergence of regularized learning in games.

M. Han, Lexington High School, Lexington, MA 02421
Email address: michaellhan@gmail.com

A. Yu, Concord Academy, Concord, MA, 01742
Email address: ashley.y.ca99@gmail.com